UDC 004.9
doi:10.31799/1684-8853-2025-2-2-15
EDN: MLUUSQ

Articles

Traffic sign recognition through the use of an Internet of Things system and deep learning

Vo Nhu Thanh[a], *PhD, Lecturer, orcid.org/0000-0001-5383-3119*
Pham Cong Thang[a], *PhD, Lecturer, orcid.org/0000-0002-6428-102X, pcthang@dut.udn.vn*
Vo Viet Truong[a], *Student, orcid.org/0009-0006-2601-0743*
Ho Van Thao[a], *Student, orcid.org/0009-0006-5016-9420*
Tran Minh Quan[a], *Student, orcid.org/0009-0001-8285-9171*
Hoang Nguyen Nhat Minh[a], *Student, orcid.org/0009-0000-1336-4426*
[a]*The University of Danang — University of Science and Technology, 54 Nguyen Luong Bang St., Danang 550000, Vietnam*

Introduction: *Automatic traffic sign recognition is an important component in modernizing traffic safety systems and minimizing related incidents.* ***Purpose:*** *To develop a Vietnamese traffic sign detection system that integrates IoT technology, cameras, sensors, and deep learning techniques.* ***Results:*** *In this study, we introduce an advanced computer vision solution that uses the neural architecture of YOLOv8 to recognize and classify traffic signs in Vietnam in real time. In addition, the system integrates the detected sign locations into Google Maps and provides a comprehensive database at all locations in urban areas. The data collection process is implemented by first capturing real-life images on the road and then combining them with the existing Vietnam traffic sign dataset from Kaggle data. To improve the reliability of the dataset, Mosaic data augmentation technique is applied. For real-time traffic sign recognition, a Raspberry Pi 4 board displays detected road signs on an HMI screen, while a custom-developed mobile application detects, recognizes, and notifies users of their locations. The trained system achieves a mAP50 value of 89% and an accuracy of 90%, which is pretty good for traffic signs detection. Moreover, the YOLOv8 model gives a precision and recall value of 89.7% and 86.8%, respectively, which is considerably higher as compared to the SVM and MSER/HOG methods.* ***Practical relevance:*** *We have succeeded in creating a real-time road sign recognition system integrated into Google Maps to meet the needs of traffic control, safety and accident reduction in the streets of Vietnam.*

Keywords *— traffic sign recognition, YOLOv8, Google Maps integration, real-time detection, road safety.*

For citation: Thanh V. N., Thang P. C., Truong V. V., Thao H. V., Quan T. M., Minh H. N. N. Traffic sign recognition through the use of an Internet of Things system and deep learning. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2025, no. 2, pp. 2–15. doi:10.31799/1684-8853-2025-2-2-15, EDN: MLUUSQ

Introduction

Traffic signs play an important role in conveying essential information, such as regulations, warnings, and instructions, which help guide drivers and reduce accidents. Recent studies have emphasized that traffic signs play a pivotal role in maintaining safety and regulating traffic, as they convey important information to drivers, pedestrians, and other road users [1]. Therefore, traffic sign recognition is essential for warning and reminding drivers to ensure road safety. However, accurate recognition of traffic signs remains a challenge due to multiple dynamic factors in real-world conditions. These include environmental variations such as extreme lighting conditions, adverse weather, and seasonal changes; physical constraints like sign deterioration and occlusions; motion-related challenges including high-speed image capture blur and vehicle vibrations; and infrastructure variations across different regions. To improve this situation, various automatic traffic sign recognition systems have been developed, each with distinct capabilities and limitations. Modern deep learning approaches like Convolutional Neural Networks (CNN) based systems [2] improve accuracy to 85–90% but demand significant computational resources, limiting their deployment on edge devices. Before the widespread adoption of neural networks, traffic sign recognition relied primarily on traditional computer vision techniques. Support Vector Machines (SVM) combined with Histogram of Oriented Gradients (HOG) [3] achieved 41.03% precision and 34.15% recall, offering computational efficiency suitable for embedded systems. Enhanced approaches using Maximally Stable Extremal Regions (MSER) with HOG features [4] improved performance to 88.75% precision and 81.35% recall. Recent commercial ADAS implementations have proposed two new systems: turn assist system and lateral clearance warning, which aim to further improve road safety in Russia [5]. Also, a multimodal mobile application called MIDriveSafely is designed to enhance road safety by detecting dangerous driving situations, providing driver feedback, offering entertainment, and enabling voice control through advanced audio-visual speech recognition technologies [6]. IoT (Internet of Things) integrated systems have emerged as a promising solution, combining edge processing for real-time detection with cloud resources for complex analysis. However, existing IoT-based implementations face challenges in seamless integration, reliable connectivity, and scalable deployment across diverse traffic environments [7]. Integrating IoT with deep learning techniques offers a promising solution for real-time traffic sign recognition. Neural network systems, specifically CNNs, demonstrate outstanding performance in traffic sign detection tasks. Within the CNN architecture, the You Only Look Once (YOLO) method stands out as an effective method for instantaneous object recognition. Among them, YOLOv8 is the latest version that provides improved accuracy, speed, and performance in detecting small objects, making it suitable for traffic sign recognition applications. Additionally, IoT devices, such as Raspberry Pi, and mobile applications, facilitate real-time collection, processing, and transmission of traffic sign data, contributing to enhanced road safety. Recent advances in artificial intelligence and deep learning have offered innovative approaches for accurate and real-time traffic sign detection [8–10]. This study investigates the capabilities of the YOLOv8 model in recognizing traffic signs and its integration with Google Maps for location mapping. Through this holistic approach, our work advances the enhancement of automated transportation networks, supporting improved road safety and operational efficiency. By integrating detected traffic sign locations with Google Maps, the system enhances practical usability, offering real-time monitoring and management of traffic signs through a user-friendly interface. This supports informed decision-making for both drivers and traffic management consultants.

Our work focuses on enhancing road safety in Vietnam by implementing a traffic sign recognition system. Utilizing a Raspberry Pi 4 combined with a webcam and a dedicated mobile application, the system aims to raise awareness and understanding of traffic signs among road users [11]. The Raspberry Pi 4 is responsible for identifying and classifying traffic signs in real-time, while the mobile application serves as the primary tool for user interaction. Upon identifying a traffic sign, the application logs its coordinates and stores the data in a database, building upon previous research in this field [12–15]. The main point of this study focuses on improving Vietnam's roadway safety by developing an automated traffic sign detection system with the four specific objectives as follows.

1. The development of an automated traffic sign identification and classification system integrates community participation in database maintenance. The system architecture facilitates user contributions while simultaneously enhancing public understanding of traffic regulations.

2. The implementation of a dynamic traffic sign database provides real-time location information through an integrated mapping system. This database serves as a comprehensive repository of traffic sign distribution data, enabling systematic analysis and improvement of traffic information management.

3. A mobile platform is built to interact and provide detailed information about traffic signs to users. This mobile platform combines mapping functionality to illustrate the location and detailed content of traffic signs, supporting users to make rational decisions when participating in traffic.

4. The integration of advanced IoT technology with continuous real-time data collection from the community allows the system to be continuously improved and upgraded, thereby increasingly enhancing the safety of people participating in traffic.

This approach allows the development of applications to support and manage traffic management infrastructure in a sustainable manner with high scalability.

Literature survey

YOLO model

There have been many studies on different traffic sign detection and recognition schemes. Early research focused on developing recognition systems using color segmentation, moment invariance, and neural networks, which have demonstrated high recognition performance, good accuracy, and low computation time [16]. Subsequent work focused on developing systems that extract and alert drivers to the content of traffic signs using text or voice commands [17]. Other important contributions include methods for detecting and recognizing traffic signs using color information and symmetry features [18] and systems based on neural networks and swarm optimization [19]. CNN models have made some important advances in the field of traffic sign recognition. One study demonstrated the effectiveness of lightweight CNN models that achieved a reduction of more than 50% in the number of required parameters while maintaining test accuracy [20]. Another study successfully implemented CNN-UM simulations on live data stream processing platforms [21], while other works proposed automated CNN systems for image classification tasks [22]. Recent works have established mathematical models to evaluate the performance of CNN systems, providing accurate estimates of power consumption and response time [23].

The YOLO framework represents a significant advancement in real-time object detection, with YOLOv8 emerging as its latest iteration. Recent applications of YOLOv8 have demonstrated particular effectiveness in specialized domains, achieving a 7.7% improvement in drone-based detection accuracy [24, 25] and 86.09% accuracy at 28 frames per second in surveillance applications [26]. For small-object detection in remote sensing applications, enhanced variants have shown superior performance compared to the base architecture [27].

The YOLOv8 implementation selected for this study utilizes key architectural features essential for traffic sign recognition: advanced data preprocessing including Mosaic augmentation, efficient feature extraction through a specialized backbone network, and a detection system optimized for varying object scales. This architecture demonstrates particular suitability for traffic sign detection due to its ability to process real-time data streams while maintaining accuracy under diverse environmental conditions [28–30].

YOLOv8 uses a comprehensive loss function that integrates box loss to optimize bounding box coordinates, class loss to improve object classification accuracy, and object-specificity loss to improve the confidence score of detected objects. Building on this foundation, the model introduces several architectural improvements, including a split-head design that separates detection and classification tasks for better specialization, and dynamic anchor assignment that adapts to different object sizes and shapes. During detection and synthesis, feature scaling improves detection across different scales. These improvements combined make YOLOv8 particularly effective for complex, real-time applications such as traffic sign recognition, where accurate detection and classification must occur simultaneously under varying environmental conditions and time constraints.

Metrics and loss function

Intersection over Union (IoU) is a crucial metric in object detection that provides a numerical assessment of the overlap between an object detector's predicted (pd) bounding box and a ground truth bounding box (gt) [31]. It defines the following important words and is a basic component for assessing a model's accuracy:

True Positive (*TP*): A detection is labeled as TP when the IoU(gt, pd) is greater than or equal to a specified threshold α, indicating a meaningful overlap between the ground truth and the prediction.

False Positive (*FP*): If the IoU(gt, pd) falls below the threshold, the detection is marked as FP. FP denotes instances where the model incorrectly predicts an object that does not align with the ground truth, indicating a false positive result.

False Negative (*FN*): FN instances occur when a ground truth object is not detected by the model, typically due to IoU(gt, pd) being below the chosen threshold α. FN represents cases where the model misses an actual object, resulting in a false negative outcome.

The area of union between the gt and pd bounding boxes is divided by the area of intersection between them to get the IoU [32]:

$$IoU = \frac{\text{Area of Intersection between gt and pd}}{\text{Area of Union of gt and pd}}.$$

The IoU value is a number between 0 and 1, where 1 denotes a perfect match or total overlap and 0 denotes no overlap. The selection of the IoU threshold α is important since it affects how detections and ground truth objects are classified and makes it possible to adjust how well the model performs. Model evaluation can be made more flexible by adjusting the threshold, which can affect the trade-off between true positives and false positives.

Precision is a measure of a model's accuracy in accurately recognizing pertinent objects. It is computed by dividing the total number of positive predictions produced by the model (TP) by the number of all Positives (TP and FP) [33]:

$$Precision = \frac{\text{True Positives}}{\text{True Positives + False Positives}}.$$

Recall measures how well a model can identify all true positives. Recall is calculated as the number of TP divided by the total number of TP and FN. Recall estimates the proportion of data that are correctly identified compared to all true positives in the data set:

$$Recall = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}.$$

Effective systems typically require high percentages in both precision and recall. However, optimizing these metrics involves balancing factors such as increasing precision often reduces recall and vice versa. This balance must be carefully adjusted during system optimization to suit operational needs.

Mean Average Precision (mAP) is a crucial parameter for assessing how well object identification models work. By combining precision and recall, it offers a thorough evaluation that gives important details about the model's correctness.

The object detection metric mAP50 uses a 0.5 overlap threshold to evaluate how well models can spot objects. When a model scores high on mAP50, it means it's doing a good job overall at finding things in images. For more thorough testing, we also look at mAP50-95, which checks detection quality across different overlap requirements from 0.5 to 0.95. This helps us understand how precisely the model can pinpoint object locations.

These measurements help us figure out if our object detection systems are actually working well in the real world. They give us useful insights about detection quality across different object types and scenarios. Systems that do well on both mAP50 and mAP50-95 tend to be more dependable, which is why they're often chosen for things like self-driving cars and security cameras.

Loss functions measure detection accuracy by comparing model predictions against ground truth data during training. Three key components form the evaluation framework:

— box loss quantifies coordinate prediction accuracy between predicted and ground truth bounding boxes;

— class loss evaluates classification accuracy within detected regions;

— defocus loss optimizes detection performance specifically for degraded image quality scenarios.

The combined loss metrics form an optimization index that guides model convergence toward optimal object localization and classification. This composite evaluation enables systematic performance assessment and model refinement across varying detection scenarios.

Mosaic augmentation

YOLOv8's training process integrates advanced data augmentation strategies to boost detection capabilities. The primary technique, Mosaic augmentation, merges four training images into one combined input. This method, which evolved through YOLOv4 and YOLOv5 iterations, exposes the model to varied object contexts and partial visibility scenarios. Such training conditions strengthen the model's ability to handle complex detection tasks in real environments.

Mosaic augmentation combines four training images into one image at a random ratio. The algorithm follows these steps [34]:

— acquire four images from data set;

— adjust dimensions for uniformity;

— combine into 4×4 matrix layout;

— crop a random image patch from the center, which becomes the final augmented image.

When applying Mosaic augmentation technique, the processed image matrices exhibit similar structural patterns as the examples shown in Fig. 1.

This algorithm enhances the sensitivity when detecting objects in heterogeneous environments, thereby reducing the dependence on the environment. This method optimizes the flexibility of the system to be able to operate on many different contexts, thereby improving the recognition accuracy. This data augmentation protocol is quite effective for datasets that exhibit spatial differences and contextual heterogeneity. However, its effectiveness decreases when applied to datasets that include text documents, objects with prominent features, or samples with invariant spatial locations.

Comparative analysis between YOLOv8 and YOLOv5 architectures in traffic sign detection applications shows a better improvement in performance (Table 1). The experimental evidence obtained demonstrates that the object detection accuracy of YOLOv8 is superior to YOLOv5 by about 2.82%. In contrast, YOLOv5 shows higher sensitivity (object detection speed), but the difference compared to YOLOv8 is quite small, about 0.54% [35].

■ ***Fig. 1.*** Examples of Mosaic augmentation applied to traffic sign images

■ ***Table 1.*** Performance comparison between YOLOv8 and YOLOv5 models

Model	Precision, %	Recall, %	Dataset size	Processing
YOLOv8	84.62	76.40	10,940	Real-time
YOLOv5	81.80	75.94	10,940	Real-time

Note: Results obtained from training on the Vietnamese traffic sign dataset.

However, based on the superior accuracy figures of YOLOv8 in other studies as mentioned above, the research team chose this network model to apply traffic sign recognition in this study.

Material and methods

The system implements two primary operational modes to effectively monitor and catalog traffic signs. In the mobile user scenario, individuals utilize a smartphone application to capture and identify traffic signs, with the system automatically processing images, determining sign types, recording GPS locations, and updating a central database. Simultaneously, fixed installations using Raspberry Pi units with cameras provide continuous monitoring at critical traffic locations, enabling real-time detection and automated database updates. The processing workflow integrates these inputs through a standardized pipeline: beginning with image acquisition from camera, followed by preprocessing for standardization, YOLOv8-based detection and classification, database updating with GPS coordinates and sign information, and finally providing user feedback through the mobile interface. This unified approach ensures comprehensive traffic sign monitoring while maintaining system accessibility and reliability. The subsequent sections provide detailed technical specifications and implementation methods for each system component.

Training model

The experimental setup utilized high-performance computing hardware and software configurations to ensure optimal training conditions. The training environment comprised a Windows 11 operating system with an Nvidia 3080Ti GPU (12 GB memory), PyTorch framework (version 1.10.0a0), and CUDA toolkit (version 12.0). The development environment was configured using Anaconda Navigator, with Ultralytics framework implementation for object detection and segmentation tasks.

Dataset preparation and processing

The dataset development followed a systematic three-phase approach:

Phase 1: Data acquisition

A comprehensive traffic sign image dataset was constructed through multiple acquisition channels, including: high-resolution digital photography; curated online repositories; manual field photography. This multi-source approach ensured diverse representation of traffic sign variations.

Phase 2: Data annotation

Image annotation was performed using specialized annotation tools (LabelImg and CVAT) to create accurate bounding box labels for each traffic sign instance. The annotation process followed strict guidelines to ensure consistency and quality of labeled data.

Phase 3: Dataset partitioning

The experimental dataset consisted of 10,940 preprocessed images obtained from Roboflow. The dataset partitioning followed standard deep learning practices with 9,540 images for training, 786 for validation, and 614 for testing. This distribution was determined based on empirical studies of deep learning model training requirements and the need to prevent overfitting while ensuring adequate validation capabilities.

YOLOv8 configuration

The post-labeling phase required systematic configuration of the YOLOv8 model parameters through a COCO-format YAML file. This configuration framework specified three critical components: the training data pathway for model learning, the validation data location for performance assessment, and the comprehensive class definitions for traffic sign detection. The structured YAML configuration ensures consistent model training conditions and facilitates experimental reproducibility. The implemented configuration structure is defined as:

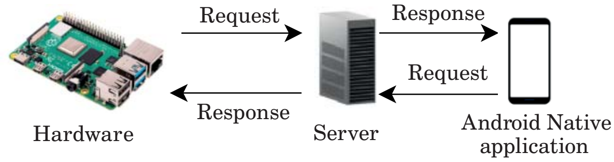
path: dataset
train: train/images
val: val/images
test: test/images
nc: 58 # Number of classes
names: ['DP.135', 'P.102', 'P.103a', 'P.103b', 'P.103c', 'P.104', 'P.106a',...]

Setting up parameters and proceeding with the training

Model implementation utilizes YOLOv8-L (large variant) architecture with input resolution of 640x640 pixels, configured with specific hyperparameters optimized for traffic sign detection. The learning rate follows a cosine decay schedule starting at 0.001, using batch size of 16, weight decay of 0.0005, and momentum of 0.937, with 3 warmup epochs. The network is structured to recognize Vietnamese traffic signs according to national standards, classifying 25 warning signs (W series)

■ ***Table 2.*** YOLOv8-L (large) training configuration parameters and their values for traffic sign detection optimization

Model parameters	Description	Value
Model	Model that we want to use	yolov8l.pt
Data	Data file	coco.yaml
Imgsz	Image size	640
Workers	The number of processes that generate batches in parallel	0
Device	Device to run training	0
Batch	The number of images processed before updating the model	2
Epochs	The number of times the learning algorithm will work to process the entire dataset	300
Patience	Epoches to wait for no observable improvement for early stopping of traing	50
Name	Folder name	yolov8-traffic-sign



■ ***Fig. 2.*** Overall system architecture

aligned with QCVN 41:2019, 20 regulatory signs (R series) following Circular 54/2019/TT-BGTVT, and 13 guide signs (G series) conforming to TCVN 7887:2018. The training follows a three-stage process: initial pre-training using ImageNet weights, domain adaptation using a COCO traffic sign subset, and final fine-tuning on our Vietnamese dataset using progressive learning rates. This configuration ensures optimal performance while maintaining compliance with local traffic standards and regulations. Through systematic parameter tuning and performance analysis, an optimal set of training parameters was established. Table 2 details these parameters, which were selected to maximize the model's traffic sign detection capabilities while maintaining computational efficiency.

The training time depends on the dataset size, the number of epochs, and the number of classes. After the training process completed, a trained YOLOv8 model is used to detect objects in real-time.

System architecture and design

The proposed system architecture integrates three fundamental components: server infrastructure, hardware platform, and software interface. Figure 2 shows this comprehensive architecture, demonstrating the interactions between components and data flow pathways.

The hardware implementation centers on a Raspberry Pi 4 computing platform, which manages image processing, display output, and camera input operations. This embedded system facilitates real-time traffic sign recognition through its integrated webcam module while maintaining continuous communication with the server infrastructure for data processing and result visualization.

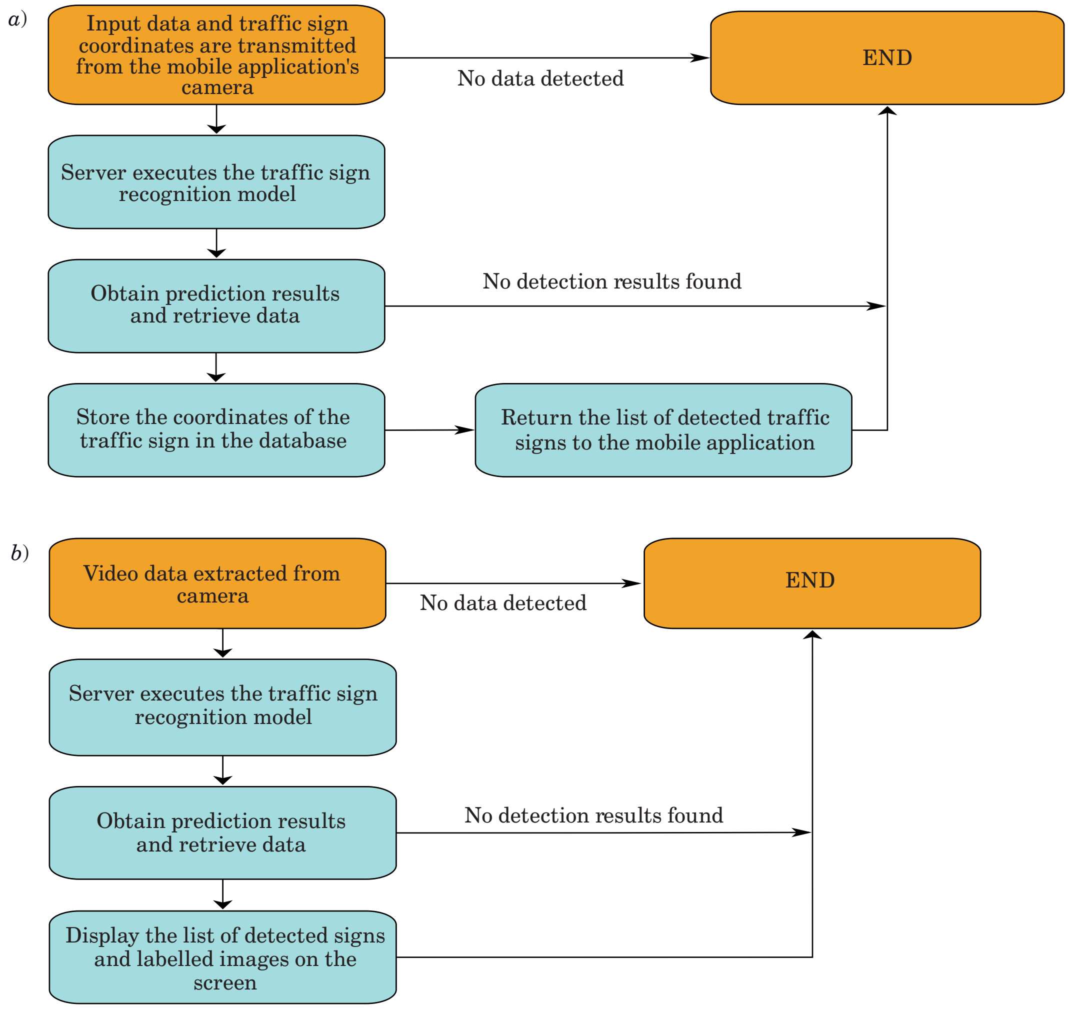
The software layer comprises an Android Native application developed in Java, implementing a client-server architecture for efficient data management and sign recognition processes. The application provides primary functionalities including user authentication, real-time sign detection, and location-based visualization through map integration. Secondary features encompass detailed sign information retrieval and supplementary data access capabilities.

The server infrastructure employs a dual-server architecture to optimize different processing requirements. A NodeJS-based server manages user authentication, session management, and geographical data storage, while a Flask-based Python server specializes in computational tasks related to traffic sign recognition. This separation of concerns ensures optimal performance for both data management and image processing operations.

Figure 3, *a*, *b* demonstrates how the web server acts as a bridge between software, hardware, and the system's AI server. Its role is to receive images collected by the mobile application for recognition processing and return the results to the mobile app while storing the location in the database. Additionally, on the hardware side, it receives streaming data to enable continuous recognition processing and displays the results on the web server interface. The diagram reveals the complete data flow and interaction pathways between all system components.

Hardware solution. The system hardware features these specifications: Dual-band networking via 2.4/5.0 GHz IEEE 802.11ac and gigabit ethernet connectivity. Four USB ports (two 3.0, two 2.0) enable camera connections. The processing unit incorporates a 40-pin GPIO interface, supports 1080p/30FPS video input, and drives a 320x480 LCD panel.

The processor handles concurrent display output and camera input operations. Visual data flows from camera sensor through the processing unit to the display module. Remote access operates through

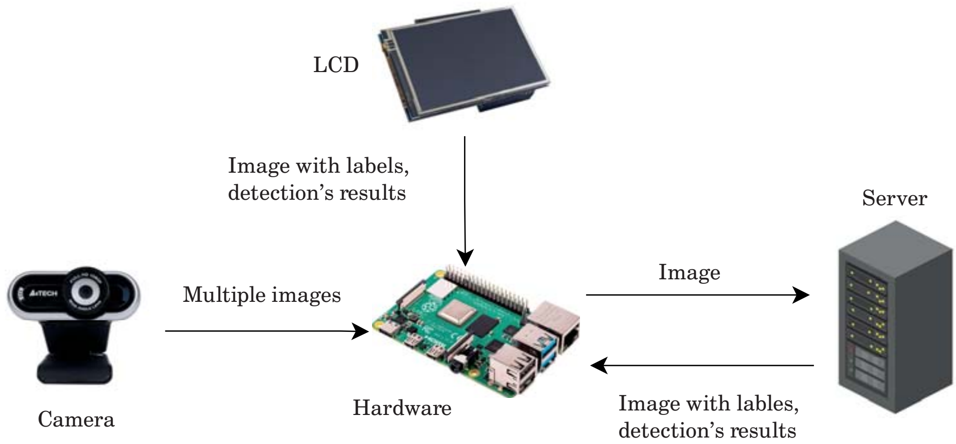


■ ***Fig. 3.*** System component interactions: *a* — web server and mobile application; *b* — web server and hardware

a NodeJS-based web portal. Captured images route to the central server for processing, with analysis results returned to the display. This configuration enables continuous visual analysis, supporting diverse research applications. Figure 4 illustrates the circuit architecture, detailing component interconnections between core processing unit, display system, and optical sensors.

Software solution. We have developed a mobile application that enables users to identify traffic signs by taking photos. This framework performs automatic sign classification, captures location data, and logs details to storage. When users submit images, the central processor analyzes them and returns comprehensive sign descriptions. The platform also enables alert parameter searches and profile customization options.

The application and interface of the traffic sign recognition system are built on Android Native (Java) software. The team chose this Android Native (Java) platform to take advantage of some of the main advantages of this platform that has been optimized for the Android operating system, providing superior performance and reliability when handling large data sets and diverse interactions between multiple devices. The platform uses the official Android language, which facilitates integration with many essential APIs, especially the Camera and Maps services, allowing full use of the system's features. Furthermore, thanks to the powerful and modern programming environment of Java and supported by continuous updates from the Android development community, access to cutting-edge technology is guaranteed to solve complex challenges. The combination of the Java platform and XML provides flexible user interface management that allows for a clear separation between logical and interface components. However, this implementation approach also poses certain challenges such as the time required for building and development is extended due to the need to build platform-specific solutions instead of using available cross-platform libraries. The platform requires quite comprehensive programming exper-



■ ***Fig. 4.*** Hardware implementation diagram

tise and extensive knowledge of the Android SDK, which can create certain difficulties for new developers. Additionally, the system scalability faces limitations such as expansion to other platforms such as iOS requires significant additional support library components compared to cross-platform alternatives. Despite these challenges, the advantages in performance, reliability, and system integration are adequate reasons to choose Android Native (Java) for this application.

Intermediary server. Figure 5 demonstrates our dual-server architecture implementation. The system employs a NodeJS server for database management and connectivity, alongside a Flask server dedicated to model storage and traffic sign recognition processing tasks.

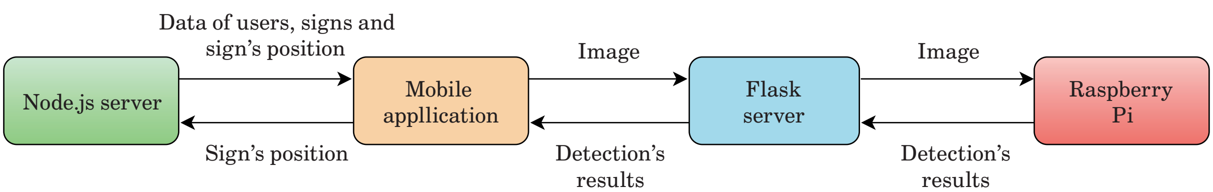
Building a server NodeJS. Used to manage all information about users, road signs, as well as all data about the location of road signs to be stored in the MySQL database.

Research has shown that Node.js demonstrates high memory efficiency compared to traditional multi-threading servers, allowing it to handle numerous concurrent requests without significant performance degradation [36]. These characteristics make it an ideal choice for managing data centrally, given its scalability, high performance, and cost-effectiveness. For the reasons mentioned above, we have built a Node.js server to handle the centralized management of data surrounding this mobile application.

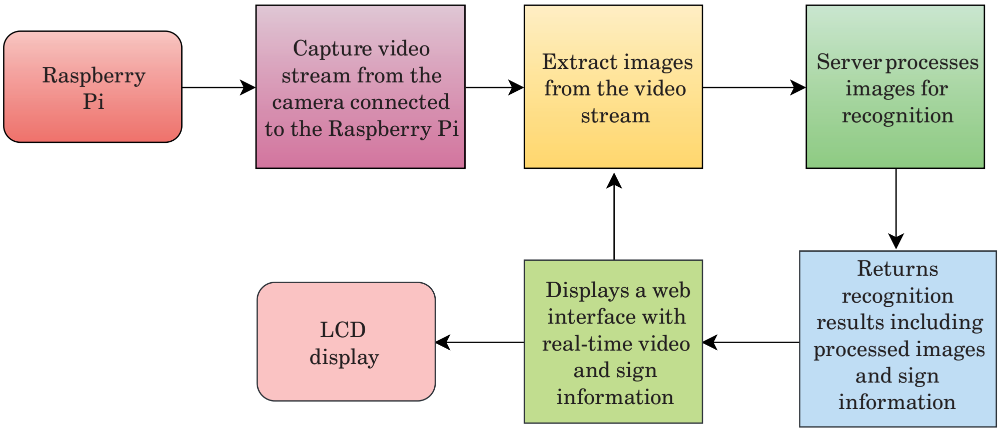
Building a server Flask. A web server was created to retrieve images, process recognition tasks, and provide a website interface capable of obtaining images from user devices for real-time recognition. The system implements Python as its core development language for neural network operations, utilizing its efficient code structure and established frameworks such as NumPy, Pandas, TensorFlow, and PyTorch. These libraries support effective implementation of complex models, making Python an ideal choice for research implementations. The system implements YOLOv8, the latest model in the YOLO family of object detection systems, which utilizes PyTorch to support traffic sign recognition with state-of-the-art performance.

Figure 6 shows the web server architecture, where Flask was selected as the framework for facilitating real-time traffic sign detection. While other robust frameworks like Django are available, research has shown that Flask's performance capabilities surpass alternatives due to its simple architecture [37]. The framework efficiently handles hundreds of requests per second without significant performance degradation, making it particularly suitable for rapid recognition tasks and real-time display of results on the web interface.

For further details, including access to the source code and comprehensive documentation, the complete repository, containing scripts, models, and additional resources, is available at (https://github.com/pacotha/DL_IOT2.git).

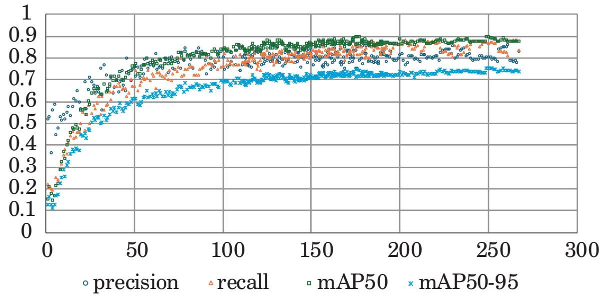


■ ***Fig. 5.*** Comprehensive system interaction flowchart



■ ***Fig. 6.*** Real-time image processing workflow on hardware components

The system employs a three-tier verification approach for sign status monitoring. Regular rescans by mobile users and fixed cameras provide updated sign condition data. A confidence scoring system flags potential sign removals when detection fails across multiple scans. Additionally, community reporting through the mobile interface allows users to flag missing or relocated signs, with automated verification triggered for reported changes.

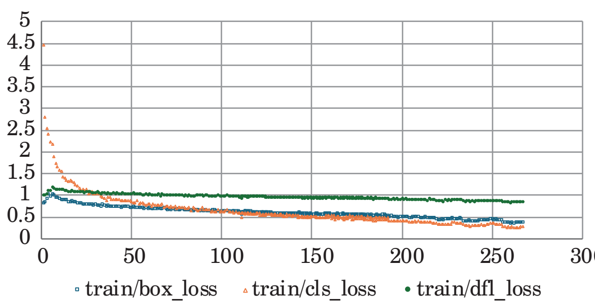


■ ***Fig. 8.*** Performance metrics evolution during training

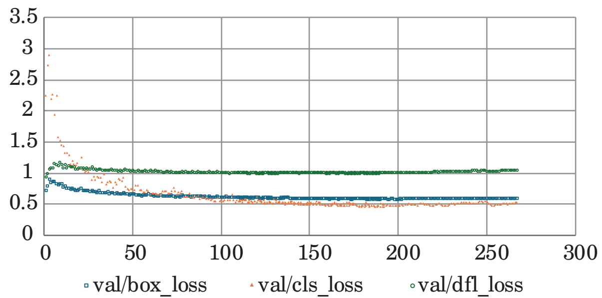
Results and discussion

The evaluation of our Traffic Sign Recognition system, implemented using the YOLOv8 architecture, high-lights its effective performance. Figure 7 illustrates the training losses for bounding box, class, and defocus over a 263-epoch training cycle. The box and class loss values, both below 0.5 after training process, indicate successful training and proficiency in object classification. Figure 8 presents the performance metrics of the YOLOv8 model, demonstrating that our system achieves a notable mAP50 score of approximately 89%. This score highlights the model's capability in recognizing objects with diverse levels of overlap across IoU thresholds ranging from 0.5 to 0.95. At the specific IoU threshold of 0.5, the system demonstrates significant accuracy, reaching nearly 90%, which validates its robust ability to accurately identify objects with moderate overlap.

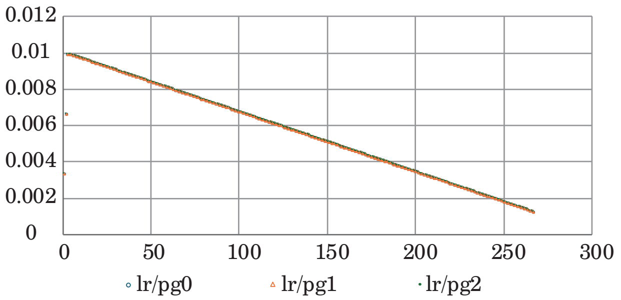
Figure 9 shows the validation phase results, where both box and class losses decreased and stabilized below 0.5, suggesting effective generalization of the model to unseen data. The defocus loss, indicated also known as dfl-loss, is a specialized component that improves object detection in challenging conditions such as blurry or unclear images. This loss remained stable around 1, indicating the model's resistance to overfitting while maintaining good



■ ***Fig. 7.*** Evolution of training losses over 263 epochs



■ ***Fig. 9.*** Validation loss trends across training epochs

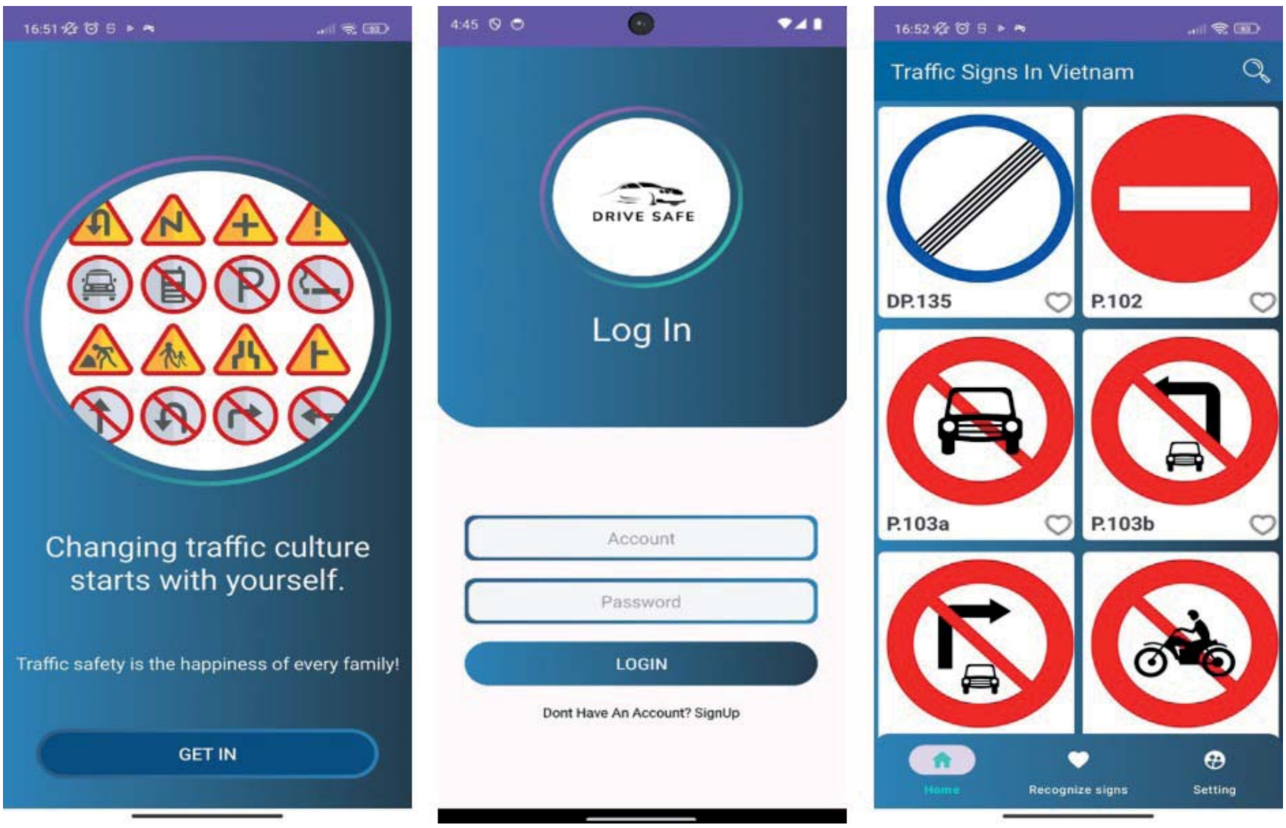


■ ***Fig. 10.*** Learning rate adaptation throughout training

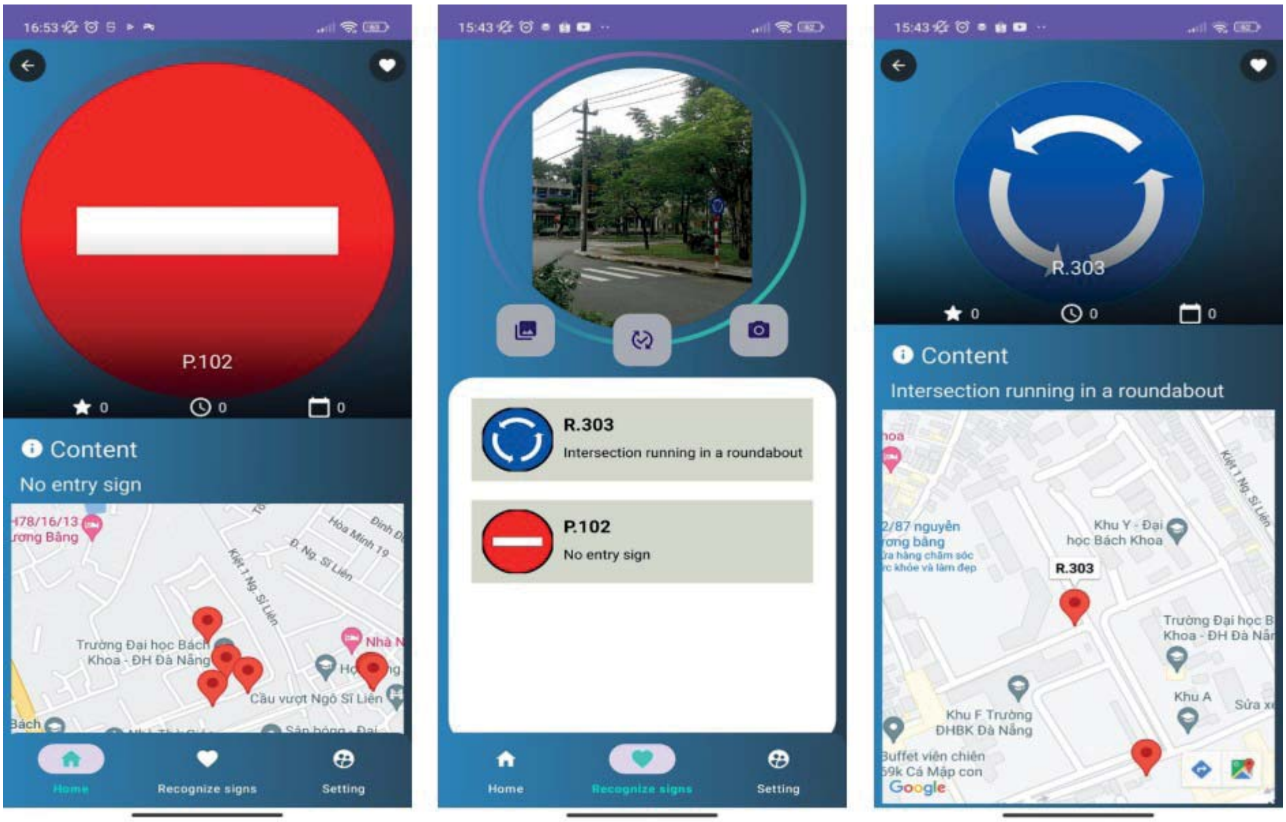
generalization capabilities. Figure 10 reveals the learning rate adaptation from 0.01 to 0.001, where the linear decrease and stabilization of training and validation losses suggest the model's convergence toward an optimal point.

The key performance metrics of the model include precision and recall, both of which are around 80%, indicating that this object detection model has a fairly good balance. This balance is important to minimize the rate of false positives while ensuring that relevant objects are correctly identified. Regarding the loss function, the box_loss decreases during training, indicating that the system effectively minimizes the error in training to predict bounding box coordinates. Similarly, the cls_loss decreases significantly during training, demonstrating the model's ability to accurately classify object types.

The mobile application interface is shown in Fig. 11. Although the positive metrics indicate that the YOLOv8 model has been properly trained, real-world evaluations show that the recognition ability is still not high. According to our survey on the street, the recognition accuracy of the model is around 70%. There are several reasons for this, including factors related to the training process and data processing. One important factor contributing to this phenomenon is overfitting, where the model focuses too much on the training data. The mobile application implements a hybrid storage approach. Frequently accessed map regions are cached locally, reducing server load and enabling offline functionality. Map data synchronizes automatically when connectivity resumes, with differential updates minimizing data transfer. User preferences determine cache size and update frequency. However, through practical observation, despite the limited recognition ability, the system still correctly places the recognized signs on Google Maps as shown in Fig. 12. This proves that our system works effectively in real-world applications.



■ ***Fig. 11.*** Mobile application user interface showcasing key features: sign recognition interface; location mapping display; user interaction components for data management



■ ***Fig. 12.*** Mobile application output displaying successful traffic sign recognition results with integrated location mapping and sign classification details

Table 3 presents a comprehensive performance comparison between the YOLOv8 model and two baseline methodologies: SVM [3] and MSER/HOG-enhanced SVM [4]. The evaluation metrics encompass recall, precision, and dataset utilization efficiency, providing quantitative insights into each model's detection capabilities in real-time traffic sign recognition.

Applying the YOLOv8 network to the system shows significant improvements in detection ability as shown by the recall data showing a 2.5x increase compared to traditional SVM methods. Specifically, the system's recall with YOLOv8 is 86.8% compared to the baseline SVM performance of 34.15%, showing a significant improvement in the ability to identify traffic signs. In addition, YOLOv8 shows better Precision than SVM, MSER/HOG, with an improvement of 39.67% (89.7% compared to 41.03% of the SWM method and 88.75% of the MSER/HOG method). This shows that YOLOv8 outperforms SVM in both object retention and classification. Compared to SVM-based MSER and HOG, YOLOv8 shows significantly higher recall value, with an improvement of 6.35% (86.8% vs. 81.45%). However, YOLOv8 has a slightly lower Precision, with a decrease of 8.05% (80.7% vs. 88.75%). This shows that SVM-based MSER and HOG have more accurate object retention, while YOLOv8 provides a higher recall.

■ ***Table 3.*** Comparative analysis of traffic sign detection performance across different methodologies: YOLOv8, SVM, and MSER/HOG

Reference	Total sign	Precision, %	Recall, %
SVM [3]	104	41.03	34.15
MSER/ HOG [4]	104	88.75	81.35
YOLOv8-L	58	89.7	86.8

Although the YOLOv8 network may show a slight decrease in precision compared to some methods, its significantly improved object retention and rapid object recognition, combined with its ability to learn efficiently from limited data, make it an effective choice for real-time traffic sign detection applications.

Conclusion

This study has successfully achieved its key objectives while advancing traffic sign recognition technology. First, our automated identification and classification system demonstrated strong technical performance, with the YOLOv8 architecture achieving an mAP50 score of 89% and 90% accuracy in controlled environments. The system enabled effective community participation through its mobile interface, though real-world accuracy averaged 70%, highlighting areas for future enhancement.

Second, we successfully implemented a dynamic traffic sign database with Google Maps integration, establishing Vietnam's first comprehensive traffic sign mapping system. The dual-server architecture, combining Node.js and Flask with Raspberry Pi edge computing, proved robust and scalable, enabling efficient data collection and real-time processing.

Third, our interactive mobile platform successfully delivered detailed sign information and educational content to users. The comprehensive mobile application facilitated user engagement and data collection, while maintaining system responsiveness and accessibility. Comparative analysis showed our approach outperformed traditional methodologies, with superior precision (89.7%) and recall (86.8%) compared to conventional SVM and MSER/HOG approaches.

Fourth, we established a scalable framework combining deep learning capabilities with community-driven data collection. This integration demonstrates the feasibility of our approach for traffic sign monitoring and maintenance. Future enhancements will focus on optimizing data augmentation techniques, improving model robustness against environmental variables, and refining real-time processing algorithms.

These achievements provide a foundation for intelligent transportation systems while identifying clear paths for future development. Initial deployment is planned for Da Nang's urban center, with phased expansion to other Vietnamese cities. The system's modular architecture enables integration with existing traffic management infrastructure and third-party ADAS systems. Future applications include real-time navigation assistance and automated traffic management. The system's architecture enables expansion to incorporate additional traffic management features and integration with broader transportation infrastructure, supporting the continued development of safer and more efficient road networks in Vietnam.

Acknowledgments

Pham Cong Thang (corresponding author) extends his gratitude to his colleagues at the Faculty of Information Technology, DUT, for their valuable feedback. The authors also sincerely appreciate the insightful comments and suggestions provided by the reviewers and editors.

References

1. Oviedo-Trespalacios O., Verity T., Barry W., Jane A. H. The impact of road advertising signs on driver behaviour and implications for road safety: A critical systematic review. *Transportation Research. Part A: Policy and Practice*, 2019, vol. 122, pp. 85–98. doi:10.1016/j.tra.2019.01.012

2. Maldonado-Bascón S., Lafuente-Arroyo S., Gil-Jiménez P., Gómez-Moreno H., López-Ferreras F. Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems*, 2007, vol. 8, no. 2, pp. 264–278. doi:10.1109/TITS.2007.895311

3. Greenhalgh J., Mirmehdi M. Real-time detection and recognition of road traffic signs. *IEEE Transactions on Intelligent Transportation Systems*, 2012, vol. 13, no. 4, pp. 1498–1506. doi:10.1109/TITS.2012.2209423

4. Wassouf Y., Korekov E. M., Serebrenny V. V. Decision making for advanced driver assistance systems for public transport. *2023 5th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)*, Moscow, Russian Federation, 2023, pp. 1–6. doi:10.1109/REEPE57272.2023.10086753

5. Ivanko D., Kashevnik A., Ryumin D., Kitenko A., Axyonov A., Lashkov I., Karpov A. MIDriveSafely: Multimodal interaction for drive safely. *Proceedings of the 2022 International Conference on Multimodal Interaction (ICMI '22)*, 2022, pp. 733–735. https://doi.org/10.1145/3536221.3557037

6. Lim K., Hong Y., Choi Y., Byun H. Real-time traffic sign recognition based on a general purpose GPU and deep-learning. *PLoS ONE*, 2017, vol. 12, no. 3, pp. e0173317. doi:10.1371/journal.pone.0173317

7. Shi Q., Abdel-Aty M. Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. *Transportation Research. Part C: Emerging Technologies*, 2015, vol. 58, pp. 380–394. doi:10.1016/j.trc.2015.07.011

8. Huangfu Z., Li S. Lightweight You Only Look Once v8: An upgraded You Only Look Once v8 algorithm for small object identification in unmanned aerial vehicle images. *Applied Sciences*, 2023, vol. 13, no. 22, pp. 12369. doi:10.3390/app132212369

9. Soylu E., Soylu T. A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition. *Multimedia Tools and Applications*, 2024, vol. 83, pp. 25005–25035. doi:10.1007/s11042-023-16451-1

10. Luo B., Sun Y., Tang Y., Gao X., Zhang F. A "Hardware-Friendly" foreign object identification method for belt conveyors based on improved YOLOv8. *Applied Sciences*, 2023, vol. 13, no. 20, Article 11464. doi:10.3390/app132011464

11. Kumar V. S., Prathibha D., Sreekanth Y., Bhanu M. Smart driver assistance system using raspberry pi and sensor networks. *Microprocessors and Microsystems*, 2020, vol. 79, iss. 8, Article 103275. doi:10.1016/j.micpro.2020.103275
12. Yang Y., Wang L., Liang J., Uddin M. Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent Transportation Systems*, 2015, vol. 17, no. 7, pp. 2022–2031. doi:10.1109/TITS.2015.2462083
13. Akple M. S., Sogbe E., Atombo C. Evaluation of road traffic signs, markings and traffic rules compliance among drivers in Ghana. *Case Studies on Transport Policy*, 2020, vol. 8, no. 4, pp. 1295–1306. doi:10.1016/j.cstp.2020.09.004
14. Ravindran R., Basu P., Bandyopadhyay S., Roy S. Traffic sign identification using deep learning. *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV, USA, 2019, pp. 318–323. doi:10.1109/CSCI49370.2019.00063
15. Siatwiko G. *Evaluating the effectiveness of the communication strategy used by the road transport and safety agency (RTSA) in promoting road safety in Lusaka urban*. Dis. the University of Zambia, 2023. Available at: https://dspace.unza.zm/handle/123456789/8090 (accessed 16 September 2024).
16. Hossain M., Hasan M., Ali M. A., Kabir M., Shawkat Ali A. S. Automatic detection and recognition of traffic signs. *2010 IEEE Conference on Robotics, Automation and Mechatronics*, Singapore, 2010, pp. 286–291. doi:10.1109/RAMECH.2010.5513177
17. Firame S., Jagtap K., Shendkar C., Dahiphale S., Sankapal S. Traffic sign detection. *Ijraset Journal for Research in Applied Science and Engineering Technology*, 2023, vol. 11, iss. IV, pp. 3438–3440. doi:10.22214/ijraset.2023.50966
18. Xu Q., Su J., Liu T. A detection and recognition method for prohibition traffic signs. *2010 International Conference on Image Analysis and Signal Processing*, Zhejiang, China, 2010, pp. 583–586. doi:10.1109/IASP.2010.5476048
19. Surinwarangkoon T., Nitsuwat S., Moore E. J. A traffic sign detection and recognition system. *International Journal of Circuits, Systems and Signal Processing*, 2013, vol. 7, no. 1, pp. 58–65.
20. He Y., Li T. A lightweight CNN model and its application in intelligent practical teaching evaluation. *2019 International Conference on Computer Science Communication and Network Security*, Sanya, China, 2019, pp. 22–23. doi:10.1051/matecconf/202030905016
21. Furedi L., Szolgay P. CNN model on stream processing platform. *2009 European Conference on Circuit Theory and Design*, Turkey, 2009, pp. 843–846. doi:10.1109/ECCTD.2009.5275115
22. Wang S., Sun L., Fan W., Sun J., Naoi S., Shirahata K., Fukagai T., Tomita Y., Ike A. An automated CNN recommendation system for image classification tasks. *2017 IEEE International Conference on Multimedia and Expo (ICME)*, Hong Kong, China, 2017, pp. 283–288. doi:10.1109/ICME.2017.8019347
23. Palit I., Lou Q., Acampora N., Nahas J., Niemier M., Hu X. Analytically modeling power and performance of a CNN system. *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Austin, TX, USA, 2015, pp. 186–193. doi:10.1109/ICCAD.2015.7372569
24. Jocher G., Chaurasia A., Qiu J. *YOLO by Ultralytics*. Available at: https://github.com/ultralytics/ultralytics (accessed 07 October 2024).
25. Wang G., Chen Y., An P., Hong H., Hu J., Huang T. UAV-YOLOv8: A small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios. *Sensors*, 2023, vol. 23, no. 16, Article 7190. doi:10.3390/s23167190
26. Shaikh S., Chopade J., Kharate G. Object classification and tracking using scaled P8 YOLOv4 lite model. *Periodica Polytechnica Electrical Engineering and Computer Science*, vol. 67, no. 1, pp. 102–111. doi:10.3311/PPee.20685
27. Ma M., Pang H. SP-YOLOv8s: An improved YOLOv8s model for remote sensing image tiny object detection. *Applied Sciences*, 2023, vol. 13, no. 14, Article 8161. doi:10.3390/app13148161
28. Lim E. A., Hong T. W., Junoh A. K. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. *Mach. Learn. Knowl. Extr.*, 2023, vol. 5, pp. 1680–1716. doi:10.48550/arXiv.2304.00501
29. Wang C.-Y., Bochkovskiy A., Liao H.-Y. M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 2023, pp. 7464–7475. doi:10.1109/CVPR52729.2023.00721
30. Akbarnezhad E. *YOLOv8 Projects #1 "Metrics, Loss Functions, Data Formats, and Beyond"*. Available at: https://www.linkedin.com/pulse/yolov8-projects-1-metrics-loss-functions-data-formats-akbarnezhad/ (accessed 16 October 2024).
31. Liu S., Qi X., Qin H., Jia J. Path aggregation network for instance segmentation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 8759–8768. doi:10.1109/CVPR.2018.00913
32. Huang Z., Yang X., Zhang Q., Li Y. Research on traffic sign detection based on improved YOLOv8. *Journal of Computer and Communications*, 2023, vol. 11, no. 7, pp. 226–232. doi:10.4236/jcc.2023.117020
33. Neha Vishwakarma. *Understanding mosaic data augmentation*. Available at: https://www.analyticsvidhya.com/blog/2023/12/mosaic-data-augmentation/ (accessed 16 September 2024).
34. Sary I. P., Andromeda S., Armin E. U. Performance comparison of YOLOv5 and YOLOv8 architectures in human detection using aerial images. *Ultima Computing: Jurnal Sistem Komputer*, 2023, vol. 15, no. 1, pp. 8–13. doi:10.19184/ultima.2023.1510001

35. Chhetri N. *A comparative analysis of Node.js (server-side JavaScript)*. Culminating Projects in Computer Science and Information Technology, 2016. Available at: https://repository.stcloudstate.edu/csit_etds/5 (accessed 16 September 2024).

36. Idris N., Foozy C. F. M., Shamala P. A generic review of web technology: Django and Flask. *International Journal of Advanced Science Computing and Engineering*, 2020, vol. 2, no. 1, pp. 34–40. doi:10.1145/1124772.1124853

УДК 004.9
doi:10.31799/1684-8853-2025-2-2-15
EDN: MLUUSQ

Распознавание дорожных знаков с использованием системы Интернета вещей и глубокого обучения

Во Ньы Тхань[a], канд. техн. наук, преподаватель, orcid.org/0000-0001-5383-3119
Фам Конг Тханг[a], канд. техн. наук, преподаватель, orcid.org/0000-0002-6428-102X, pcthang@dut.udn.vn
Во Вьет Чыонг[a], студент, orcid.org/0009-0006-2601-0743
Хо Ван Тхао[a], студент, orcid.org/0009-0006-5016-9420
Чан Минь Куан[a], студент, orcid.org/0009-0001-8285-9171
Хоанг Нгуен Нят Минь[a], студент, orcid.org/0009-0000-1336-4426
[a]Университет Дананга — Университет науки и техники, Нгуен Лыонг Банг ул., 54, Дананг, 550000, Вьетнам

Введение: автоматическое распознавание дорожных знаков является важной частью модернизации систем безопасности дорожного движения и минимизации связанных с ним инцидентов. **Цель:** разработать систему распознавания дорожных знаков Вьетнама с интеграцией IoT-технологий, камер, датчиков и методов глубокого обучения. **Результаты:** представлено передовое решение компьютерного зрения, использующее нейронную архитектуру YOLOv8 для распознавания и классификации дорожных знаков Вьетнама в реальном времени. Система интегрирует местоположение обнаруженных знаков в Google Maps и предоставляет обширную базу данных местоположения всех знаков в городских районах. Сбор данных осуществляется путем первоначальной съемки изображений с реальных дорог и их объединения с существующим набором данных дорожных знаков Вьетнама из Kaggle. Для повышения надежности набора данных применяется техника аугментации Mosaic. Для распознавания дорожных знаков в режиме реального времени плата Raspberry Pi 4 отображает обнаруженные дорожные знаки на экране HMI, в то время как специально разработанное мобильное приложение обнаруживает, распознает и уведомляет пользователей об их местоположении. Обученная система достигает 89 % значения mAP50 и 90 % точности, что довольно хорошо для распознавания дорожных знаков. Кроме того, модель YOLOv8 демонстрирует 89,7 % точности (Precision) и 86,8 % полноты (Recall), что значительно выше по сравнению с методами SVM и MSER/HOG. **Практическая значимость:** созданная интегрированная в Google Maps система распознавания в реальном времени дорожных знаков удовлетворяет потребности регулирования дорожного движения, обеспечения безопасности и снижения аварийности на улицах Вьетнама.

Ключевые слова — распознавание дорожных знаков, YOLOv8, интеграция в Google Maps, обнаружение в реальном времени, безопасность на дорогах.

Для цитирования: Thanh V. N., Thang P. C., Truong V. V., Thao H. V., Quan T. M., Minh H. N. N. Traffic sign recognition through the use of an Internet of Things system and deep learning. *Информационно-управляющие системы*, 2025, № 2, с. 2–15. doi:10.31799/1684-8853-2025-2-2-15, EDN: MLUUSQ

For citation: Thanh V. N., Thang P. C., Truong V. V., Thao H. V., Quan T. M., Minh H. N. N. Traffic sign recognition through the use of an Internet of Things system and deep learning. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2025, no. 2, pp. 2–15. doi:10.31799/1684-8853-2025-2-2-15, EDN: MLUUSQ